Technical white paper

HP ProLiant DL980 G7を使った 26×26格子グラフのパス総数数え上げ

社会や経済の変革の起爆剤となるようなイノベーションを生み出すため、独創的な発想で世界レベルの基礎研究に取り組んでいる研究者を選定し、その研究活動を全面的に支援する独立行政法人科学技術振興機構（JST）の基礎研究推進事業「ERATO」。北海道大学大学院情報科学研究科の湊 真一教授は、計算機科学の分野でこうした一人として選ばれた研究者であり、湊教授をリーダーとした「ERATO 湊離散構造処理系プロジェクト」では、アルゴリズム技術の研究を切り口として、集合や記号、グラフ、組み合わせといった様々の数学的な離散構造をコンピューターを使って極めて高速に処理するための基礎技術基盤構築を目指している。その取り組みの一つとして、n×nの格子グラフの対角2点間を、同じところを2度通らずに結ぶパスの総数を効率的に、現実的な時間内で数え上げるというパス列挙問題の研究も実施。2013年11月には、HP ProLiant DL980 G7を使い、n=26のケースにおけるパス総数数え上げを世界で初めて達成した。このホワイトペーパーでは、この経緯を紹介する。

1. 離散構造処理に伴う「組み合わせ爆発」の恐ろしさ

一般に離散構造に何らかの処理を行う場合、例えば、論理処理における「false」「true」といったように、場合分け（分岐）処理を必要とすることが多い。このため、分岐のポイントが増えるにつれてその解の数は指数関数やべき乗で急速に増えていってしまう。こうした現象は「組み合わせ爆発」と呼ばれている。

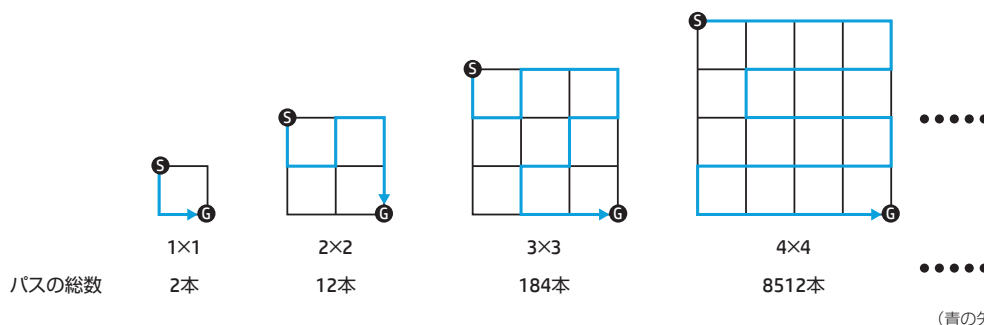
「組み合わせ爆発」の凄まじさを端的に示す例として、n×nの格子グラフで左上のスタートから対角右下のゴール点まで、同じところを2度通らずにたどり着くすべてのパスを数え上げるという問題が挙げられる（図1）。最短のパスだけでなく遠回りも許す、という条件ですべてのパスを数えていくと、1×1（n=1）では2本、2×2（n=2）では12本、3×3（n=3）では184本と増えていき、わずか3の違いしかないn=6では5億7578万0564本、n=7では7893億6005万3252本と爆発的に増えていってしまうのだ。最短で結ぶパスだけであれば、高校時代に習った組み合わせの公式「${}_{2n}C_n$」から、簡単に総数を求めることができる。しかし、遠回りを許すケースでパス総数を導き出す公式などは今のところ見つかっておらず、ひたすら数え上げるしか方法がない。

こうした「組み合わせ爆発」の凄まじさについて、最近のネット上では「おねえさん問題」として広く知られるようになっている。きっかけは、同プロジェクトの取り組みを一般の人々にも広く認知してもらうため、2012年8月～2013年4月に日本科学未来館で行われた展示用に作成されたアニメーションだ。格子グラフのnが大きくなるにつれて、パス総数を数え上げるのがいかに大変になるか、時間も果てしなくかかるのかをユーモラスに描いた作品で、タイトルは「おねえさんといっしょ！　みんなで考えてみよう」。このタイトルに由来したものだ。監修は湊教授をはじめとするプロジェクトメンバーが担当。YouTubeやニコニコ動画でも公開・転載されたことで優に100万回を超える再生回数を数え、サイエンス系動画では異例の人気を博している。

図1. 格子グラフのグリッドが増えるにつれてパスの総数は爆発的に増加

	1×1	2×2	3×3	4×4	……
パスの総数	2本	12本	184本	8512本	……

（青の矢印はパスの1例）

インテル® Xeon® プロセッサー E7 ファミリー

2. 処理の回数をアルゴリズムで劇的に圧縮する

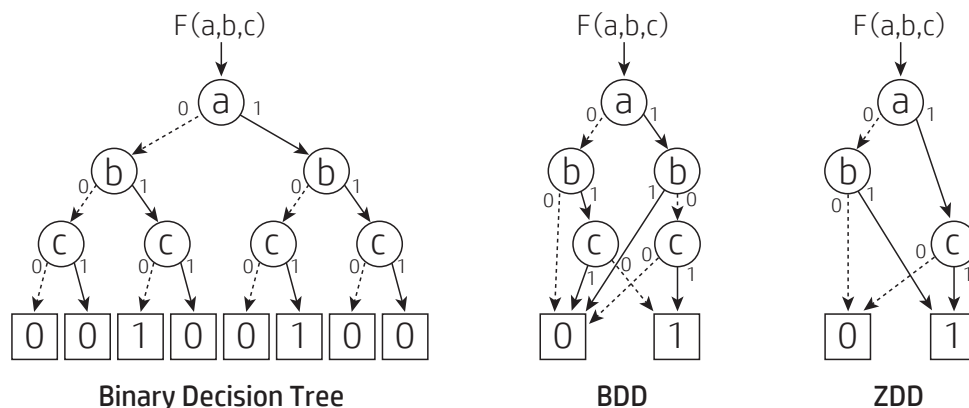
格子グラフの対角を結ぶパス総数を単純に数えるだけにも見える「おねえさん問題」だが、その応用面を考慮すると、すべてのパスを調べ上げるパス列挙問題の解決は極めて重要な意味を持つ。格子グラフの図形を通信ネットワークや道路ネットワークなどに置き換えてみると、それがわかるだろう。すべてのパスを調べ上げておくことにより、例えば、通信経路の最適化、目的地途中で起きた渋滞の回避ルート選択といったことが容易になる可能性があるからだ。つまり、膨大な処理が必要となるパス列挙問題を現実的な時間内で解くことは、実社会で想定される様々な課題を解消するうえで大きな意義があるといえる。

ただし、大規模なパス列挙問題を現実的な時間内で解くことは、コンピューターの処理能力が大幅に向上した今日においても、容易なことではない。また、極めて処理能力の高いスーパーコンピューターを使おうとすると、今度はコストが負担しきれなくなる。ここで鍵を握るのが、処理の計算手順・戦略を記述するアルゴリズムだ。アルゴリズムを工夫することで演算の回数を減らすことができれば、処理の高速化が可能になる。

この点で、同プロジェクトは強力な武器を持っている。湊教授が1993年に世界で初めて考案し、命名した「ゼロサプレス型BDD（ZDD：Zero-suppressed BDD）」がこれだ。

離散構造に関する最も基本的なモデルの一つに論理関数があるが、一般に論理関数の値をすべての変数について場合分けした結果は「二分決定木（Binary Decision Tree）」として表現することができる（図2）。コンピューターの立場から見れば、分岐の数は処理の回数を意味する。

処理回数を減らすための工夫として、BDD（Binary Decision Diagram）というデータ構造を用いたアルゴリズムが、1986年に米国で考案された。湊教授が考案したZDDは、このBDDを集合データの処理に特化させたもので、特に「疎な組み合わせの集合」、例えば、1万アイテムの商品から10を選び出すといったように、巨大な母集団から非常に少数だけを抽出するようなケースで、処理回数の極めて大きな圧縮効果が得られる。扱う問題にもよるが、圧縮の効果はBDDで数十～数百分の1、ZDDを使えばさらにBDDの100分の1程度の圧縮が可能になる。

図2. 二分決定木の例、および対応するBDDとZDD

出典：「おねえさんの問題」の最先端 –YouTube動画と世界記録-, 湊 真一, 情報処理 Vol. 54 No. 11 Nov. 2013, P1152-1159

3. 格子グラフのパス総数数え上げで世界記録を2012年にも達成

パス列挙問題に対するZDDの有効性を検証するため、同プロジェクトの研究員である岩下洋哲氏を中心とする開発チームは、「おねえさん問題」としても知られることになったn×n格子グラフのパス総数の数え上げ研究にも取り組んでいる。その成果として、いくつもの世界初という記録を積み上げてきた。

まず、2012年には汎用的なプログラムを使って、n=21のケースまで数え上げに成功。同年9月上旬にオンライン数列百科事典サイト「The On-Line Encyclopedia of Integer Sequences（OEIS）」に申請し、公式に登録された。それまでに登録されていた記録はn=19であり、一気に記録を2段階更新している。ちなみにn=21のパス総数は3.137×10^{106}本という巨大なもの。

その後、n×n格子グラフに特化したアルゴリズムの工夫、「中国人剰余定理」を用いることによる超巨大数記憶のためのメモリー量節約、マルチコアを使った並列化、などの処理高速化のための改善手法を開発することで、より大きな格子グラフでもパス数え上げに挑戦。2013年4月には、n=25のケースまでの数え上げに成功した。n=24までの数え上げはノルウェーの大学チームによって一歩先んじてOEISに登録されていたため、改めての世界記録達成となった。

この数え上げには、32コア、主記憶1.5TBのサーバーを利用。約1週間、このサーバーを占有することで達成した。さらに1ステップ拡張したn=26での数え上げにもチャレンジしようということになったが、このサーバーで同様に計算させようとすると、約3週間もの長期間にわたって占有しなくてはならないことが予想された。このため、さらに高性能なサーバーの利用を検討。インテル® Xeon® プロセッサー E7ファミリーを8基（80コア）、メモリーも最大4TBまで搭載できるHP ProLiant DL980 G7で記録更新を目指すことにした。

インテル® Xeon® プロセッサー
E7 ファミリー

4. HP ProLiant DL980 G7を使った予備検証を実施

n=26での数え上げに先立ち、HP ProLiant DL980 G7の巨大なリソースを最大限活用できるよう、チームではすでに成功していたn=25の計算の一部を実行することにより事前検証を行った。

事前検証に使用したHP ProLiant DL980 G7のスペックは次のようなものだ。プロセッサーはインテル® Xeon® プロセッサー E7-4870 2.4GHz 8基／80コア、搭載メモリー量は4TB。OSにはRed Hat Enterprise Linux 6.3、計算プログラムはn=25で使用したものを、その時と同様、gccでコンパイルした。

n=26の数え上げでは、使用するメモリー量を節約するため、中国人余剰定理に基づいて計算を10回に分割。そうすると各回のメモリ使用量は約1.4TBになる。そこで、処理の高速化を図るため、HP ProLiant DL980 G7のプロセッサーおよびメモリーを二つに分割し、見かけ上40コア、2TBのサーバーを仮想的に2台用意し、2回分の計算を並列に処理するという方法で行うことを想定していた。

このため、以下の三つの稼働方法で、スレッド数を徐々に増やしていったときの演算時間を計測（図3）。どの方法がこうした想定に最も適するかを検討した。

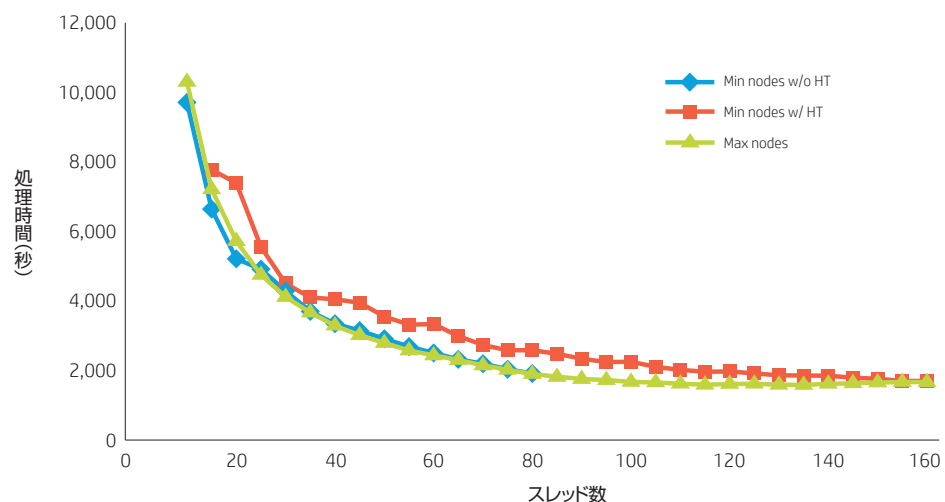
1）Min nodes without HT（図3のチャートの青プロット）
　ジョブをノード0のスレッドから順番に割り当てていくケース。ただし、ハイパースレッディングは使用しない。

2）Min nodes with HT（図3のチャートの赤プロット）
　1）同様、ノード0のスレッドからジョブを順番に割り当てていき、ハイパースレッディングも使用する。

3）Max nodes（図3のチャートの緑プロット）
　すべてのノードを均等に使用したケース。81スレッド目からはハイパースレッディングを使用する。想定する使用方法からは外れるが、負荷分散の影響を確認する意味で実施した。

図3のチャートから、次のような評価を行った。

- いずれのケースでも、スレッド数の増加とともに処理能力が確実にスケールする
- 1）では、使用するノード数が少ない領域で特に高いパフォーマンスを示す。しかし、ハイパースレッディングを使用しないため、リソースを最大限活用することにつながらない。
- 2）では、スレッド間に競合が発生しているためか、同じスレッド数での結果を比較すると 1）に対して若干のパフォーマンス低下が見られる。しかし、同じ40コアを使用した 1）の40スレッド処理と 2）の80スレッド処理を比較すると、ハイパースレッディングがパフォーマンス向上に貢献していることが わかる。

図3. HP ProLiant DL980 G7を使ったn=25格子グラフでの事前検証結果

5. n=26のパス総数数え上げを1週間で達成し、自らの世界記録をさらに更新

前述した事前検証の評価結果から、n=26のパス総数数え上げでは、ノード0のスレッドからジョブを順番に割り当て、ハイパースレッディングも使用する、という2）の稼働方法を選択。搭載メモリーを4TBに拡張し、HP ProLiant DL980 G7を4プロセッサー（40コア、80スレッド）／メモリー2TBのサーバー2台に見かけ上分割して並列処理させることで数え上げを実行した（図4）。

要した時間は約1週間。従来のサーバーで見込まれた約3週間と比べ、1/3の時間短縮が図れた。算出されたパス総数は約1.736×10^{163}本。この数字が最後の1桁まで正確かどうかはまだ検証されていないが、2013年11月18日、新しい世界記録としてOEISのサイトに公式登録された（図5）。

今回の世界記録の達成に貢献したHP ProLiant DL980 G7に対する印象を、チームの岩下洋哲氏はこう語る。「プロセッサー数／コア数を多く搭載できるサーバーの中には、バスの競合がボトルネックとなり、パフォーマンス低下を招いてしまうものも見受けられます。計算中の稼働状況を観察していた限りでは、HP ProLiant DL980 G7は非常に高いパフォーマンスを発揮してくれました」。

インテル® Xeon® プロセッサー
E7 ファミリー

また、湊教授も「格子グラフのパス総数を数え上げる処理は、ある意味、サーバーとしての底力が試されるテーマ。ランダムなメモリーアクセスを高速に行えるか、多くのコアが稼働してもバス競合などを起こさないかなど、サーバー全体のデザインが問われる処理でしたが、HP ProLiant DL980 G7は見事に期待に応えてくれました」と評価する。

図4. n=26の格子グラフのパス総数数え上げで使用したシステム構成

ハードウェア:HP ProLiant DL980 G7
搭載プロセッサー:インテル® Xeon® プロセッサー E7-4870 2.4GHz 8基/80コア
搭載メモリー:4TB
搭載記憶領域:900GB HDD×8台　合計7.2TB
OS:Red Hat Enterprise Linux 6.3

図5. OEISサイトに掲載されているn=26までのパス総数

0	1
1	2
2	12
3	184
4	8512
5	1262816
6	575780564
7	789360053252
8	3266598486981642
9	41044208702632496804
10	1568758030464750013214100
11	182413291514248049241470885236
12	64528039343270018963357185158482118
13	69450664761521361664274701548907358996488
14	227449714676812739631826459327989863387613323440
15	2266745568862672746374567396713098934866324885408319028
16	687454456091499315876315631324892 3282458794596809945 7285419306
17	6344814611237963971310297540795524400449443986866480693646369387855336
18	1782112840842065129893384946652325275167838065704767655931452474605826692782532
19	1523344971704879993080742810319229690899454255323294555776029866737355060592877569255844
20	3962892199823037560207299517133362502106339705739463771515237113377010682364035706704472064940398
21	31374751050137102 7204205381373822145131033121936987236530613519913464333793893857939655769922460213164638 68
22	755970286667345339661519123315222619353103732072409481167391410479517925792743631234987038883317634987271171404439792
23	55435429355237477009914318489061437930690379970964331332556958646484008407334885544566386924020875711242060085408513482933945 72
24	123717122312070647583387448626735708323730419890129435 39678727080484951695515930485641394550792153037191858028212512280926600305813867910 94
25	84029748578811334710070837454368091272960542937753835498247426239370284978982152569291785770839709601216256025060273165497184020649404997837560424740 8
26	173699315862792729311754404212364989003722295882881406046637037209103424132761347627892181934980061070822962231433804913482900272193112962770873889085390810 8 906396

出典:https://oeis.org/A007764/b007764.txt

6. パス総数数え上げの手法は、道路網や配電網などの解析・制御などにも応用可能

今回、格子グラフのパス総数数え上げの記録を更新したことにより、現実の社会が抱える様々な課題を計算によって解決できる可能性がまた一段と広がった、と湊教授は語る。「格子グラフでのパス総数数え上げが象徴する、コンピューターを使った離散構造処理は、道路網や鉄道網、電力網などを解析したり制御したりするために広く活用することが可能です。例えば、格子グラフを道路網に置き換えると、n=25の時には考慮できる道路の数は1300本です。これがn=26になると約1400本まで広がるのです。このときの組合せの数は2^{1300}から2^{1400}に増大するので、その差は非常に大きいといえるでしょう」。

離散構造処理は、格子グラフからすぐに連想できる分野だけでなく、防災とも関連した避難所の配置、このところ話題になることも多い選挙の区割りなどにも応用が可能だ。さらには遺伝子配列に関連した生命科学、ビッグデータからのデータマイニング、機械学習などの進歩にも大きく貢献できる可能性がある。
「こうした広範な社会的な課題へ適用の範囲を広げていくためにも、ベースとなるコンピューターのさらなる進化を期待すると同時に、これまでにない新しいアルゴリズムや計算手法を開発する努力を今後も続けていく必要があります」と、湊教授は決意を新たにしている。

お問い合わせはカスタマー・インフォメーションセンターへ
03-5749-8328 月〜金 9:00〜19:00　土 10:00〜17:00(日、祝祭日、年末年始および5/1を除く)
機器のお見積もりについては、代理店、または弊社営業にご相談ください。
HP ProLiantに関する情報は http://www.hp.com/jp/proliant

〒136-8711 東京都江東区大島2-2-1